Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Matjaž Čuk

Detekcija položaja krogel pri biljardu s pomočjo računalniškega vida

diplomsko delo

univerzitetni študij računalništva in informatike, smer računalniški sistemi

Mentor: doc. dr. Luka Šajn

Ljubljana, 2014

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Študent naj razišče osnove računalniškega vida za detekcijo položaja krogel pri biljardu. Zaradi distorzije leč kamere, s katero so slike zajete, so slike popačene. Za pravilno analizo naj študent poskuša uspešno odstraniti to popačitev. Nad slikami, namenjenimi detekciji, naj poskuša razviti algoritem za detekcijo krogel. Algoritem naj detektira lokacije posameznih krogel na mizi. Rezultati analize posamezne slike naj se zapišejo v datoteko.

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Matjaž Čuk, z vpisno številko **63070063**, sem avtor diplomskega dela z naslovom:

Detekcija položaja krogel pri biljardu s pomočjo računalniškega vida

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Luka Šajna,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 19. junija 2014 Podpis avtorja:

Rad bi se zahvalil vsem, ki so mi skozi leta študija stali ob strani.

Zahvaljujem se mentorju doc. dr. Luku Šajnu za pomoč in vodenje pri opravljanju diplomskega dela. Rad bi se tudi zahvalil Andreju Koresu, ki mi je svetoval pri izboljšanju algoritma.

Zahvaljujem se tudi podjetjema Klika d.o.o in Sportradar AG, v katerih sem deloma razvil programski del naloge in dobil mnogo koristnih predlogov pri implementaciji algoritma. Prav tako se zahvaljujem podjetjema Sports Statistics & Information Systems Ltd in World Snooker Ltd, ki sta dovolila uporabo slik za namen te diplomske naloge.

Kazalo

Povzetek

Snooker je igra, ki je izjemno popularna v Združenem kraljestvu, v zadnjih letih pa pridobiva na obožavalcih tudi v drugih državah. Igralna miza je skoraj dvakrat večja kot pri navadnem biljardu, prav zato pa je s prostim očesom še težje natančno določiti lokacijo krogle.
V diplomskem delu smo si za cilj zadali prepoznati krogle na igralni mizi in določiti njihove točne lokacije s pomočjo računalniškega vida. Razvili smo algoritem, ki zazna nad 99% krogel in določi lokacijo na nekaj centimetrov natančno. Z nekaj popravki je algoritem možno prilagoditi tudi za detekcijo krogel pri drugih tipih biljarda.

Ključne besede : detekcija krogel, snooker, računalniški vid, OpenCV, detekcija krogov

Abstract

Snooker is a very popular game in the United Kingdom, but is also gaining in popularity in other countries. Table size in snooker is almost two times the size of a pool billiards table. Determining the ball location with the naked eye is thus even more difficult.
The goal of this diploma thesis is to detect snooker balls and determine their position on the table with the use of computer vision. We developed an algorithm that detects over 99% of the balls and determines the ball location to centimetre accuracy. With a few modifications, it would also be possible to use our algorithm for ball detection on any billards table.

Key words : ball detection, snooker, computer vision, OpenCV, circle detection

Poglavje 1

Uvod

Izbrani problem te diplomske naloge je najti krogle na snooker mizi in čim natančneje določiti njihovo pozicijo s pomočjo računalniškega vida. Tako lahko izpišemo točne lokacije krogel, ki bi jih s prostim očesom sicer zelo težko določili. Rezultat našega dela je lahko tudi osnova za nadaljnje delo določitve barv krogel. Problem iskanja krogel lahko poenostavimo na iskanje krogov, saj obdelujemo 2D zajem stanja na mizi. Z detekcijo krogov so se raziskovalci že ukvarjali in je o tem že veliko napisanega. Rezultati so različni, s pravim pristopom pa je zadani cilj vsekakor dosegljiv. Igralna miza je v našem primeru posneta s strani in je tako detekcija krogel težja, saj moramo upoštevati različne velikosti krogel in pozicijo krogle preslikati v koordinate, ki bi jih imela krogla v ptičji perspektivi. Na precej slikah pogled na mizo ovira igralec, ki sloni čez mizo. Ta primer ne bi bil tako pogost, če bi bila kamera postavljena naravnost nad mizo. Tudi s perfektno detekcijo v primerih, kjer igralec z nagibom čez mizo zakriva krogle, ne moremo določiti vseh krogel, saj preprosto niso vidne. Za detekcijo vseh krogel bi v takih situacijah potrebovali posnetke iz različnih strani mize. Mi žal te možnosti nismo imeli, smo se pa odločili, da slik s igralcem preko mize ne izločimo iz analize. V takih primerih je moral algoritem izluščiti informacijo o vidnih kroglah, na samem igralcu pa ni smel zaznati lažnih krogel.
Za dosego cilja smo izbrali programski jezik Java ter odprtokodno knjižnico

OpenCV(**Open** Source **C**omputer **V**ision) [1], ki je spisana v C++ programskem jeziku, vendar ima od leta 2013 naprej poleg ostalih tudi javanski vmesnik. To omogoča preprosto uporabo v Javi s hitrostjo C++. Implementiranih ima mnogo algoritmov, ki so pogosto uporabljeni v računalniškem vidu in tako omogoča hiter razvoj.

V diplomskem delu bomo najprej opisali samo igro snooker in njena pravila, da bralec dobi osnovno predstavo o igri. V naslednjem poglavju bomo raziskali množico slik, nad katerimi smo poganjali algoritem. Sledi opis predpriprave, ki je potrebna, da lahko naš algoritem iskanja krogel uspešno deluje. Sledi glavni del, ki je opis delovanja našega algoritma. Na koncu bomo predstavili še dosežene rezultate in naredili primerjavo s drugim pogosto uporabljenim algoritmom za iskanje krogov.

Poglavje 2

Snooker

2.1 Zgodovina igre snooker

Začetki biljarda segajo v 14. stoletje. Snooker [2] je v primerjavi z biljardom precej mlajši. V 19. stoletju so britanski častniki v Indiji za kratkočasenje zelo radi igrali biljard. Med njimi je bil tudi Neville Chamberlain, ki je ob obilici časa začel eksperimentirati z različnimi variacijami biljarda [3]. Piramidnemu biljardu, ki vsebuje rdeče in črno kroglo, je leta 1875 dodal še krogle drugih barv. Najprej je dodal rumeno, zeleno in roza, čez nekaj let pa sta sledili še modra in rjava. Izraz *snooker* je bil takrat v uporabi kot vzdevek za kadete prvega letnika kraljeve vojaške akademije. Ko je nek častnik zgrešil kroglo, ga je Chamberlain označil za „snookerja“ v tej igri. Izraz se je hitro prijel in tako je ta igra dobila svoje ime. Leta 1885 je v Indijo pripotoval takratni najboljši igralec biljarda John Roberts, kjer ga je Chamberlain seznanil s snookerjem. Po tem je hitro dobival na popularnosti. Prvo uradno tekmovanje je potekalo leta 1916 [4]. Leta 1927 je Joe Davis pomagal ustvariti prvi profesionalni turnir *World Snooker Championship*, kjer je tudi zmagal. Snooker je postopoma postajal le še bolj priljubljen in dandanes potekajo turnirji z veliko televizijsko gledanostjo in visokimi denarnimi nagradami [5].

2.2 Pravila igre snooker

Snooker je igra podobna običajnemu biljardu. Igra se na mizi, pokriti s posebnim blagom, ki je običajno zelene barve. Pri najbolj pogosti igri biljarda, osmici, je na začetku na mizi 15 različno obarvanih in označenih krogel. Poleg teh petnajstih je v igri še bela krogla. Igralec mora s palico zadeti belo kroglo, z njo poskušati zadeti ostale krogle in jih zbiti v žepe igralne mize. Igrata dva igralca, lahko pa tudi dve ekipi. Vsakemu igralcu pripada sedem krogel njegove skupine, črna krogla pa jima je skupna. Zmaga tisti, ki prvi potopi svojih sedem krogel in na koncu še skupno črno kroglo.
Snooker je v marsikaterem pogledu različen od navadnega biljarda. V začetni poziciji je na mizi 22 krogel. Krogle so različnih barv in imajo različno vrednost. Enako kot pri navadnem biljardu lahko igralec udari samo belo kroglo s katero poskuša „potopiti“ ostale krogle. Na mizi je 15 rdečih krogel, vsaka je vredna 1 točko. Ostane še 6 krogel, ki si sledijo po vrednosti točk : rumena (2 točki), zelena (3 točke), rjava (4 točke), modra(5 točk), roza(6 točk) in črna(7 točk). Bela krogla je vredna 0 točk in njena potopitev velja kot prekršek. Začetni razpored je viden na sliki 2.1.

Slika 2.1: Začetni razpored.

Dokler so na mizi rdeče krogle, mora igralec najprej potopiti rdečo kroglo, nato pa kroglo druge barve. Rdeče krogle se ob potopitvi odstranijo iz igre, krogle ostalih barv pa se vračajo nazaj na začetno mesto, dokler ne zmanjka rdečih krogel. Ko se to zgodi, mora igralec potopiti ostale krogle v vrstnem redu od točkovno najmanj do največ vredne, torej rumeno, zeleno, rjavo, ... Zmaga tisti igralec, ki ima na koncu več točk. Igralec je na potezi, dokler potaplja krogle v veljavnem zaporedju. Če zgreši, potopi napačno kroglo ali potopi več kot eno kroglo naenkrat, je na potezi drugi igralec. Vseh možnih točk, pridobljenih s potopitvijo krogel, je 147. Trenutno rekord za največkrat doseženih 147 točk drži Ronnie O'Sullivan, ki mu je to uspelo dvanajstkrat [6]. Točke lahko igralec dobi tudi z napakami nasprotnika. Med točkovane napake spadajo udarec krogle napačne barve, potopitev bele krogle, potopitev več kot ene krogle ali pa v primeru, ko se bela krogla po udarcu ne dotakne nobene druge krogle. Napake so vredne od 4 do 7 točk. Tako je končno število točk lahko tudi večje od 147. Če igralec ugotovi, da ima nasprotnik v trenutni igri preveč točk in ga ne bo mogel dohiteti, lahko igro preda, kar je tudi pogosta praksa.

Poglavje 3

Slike

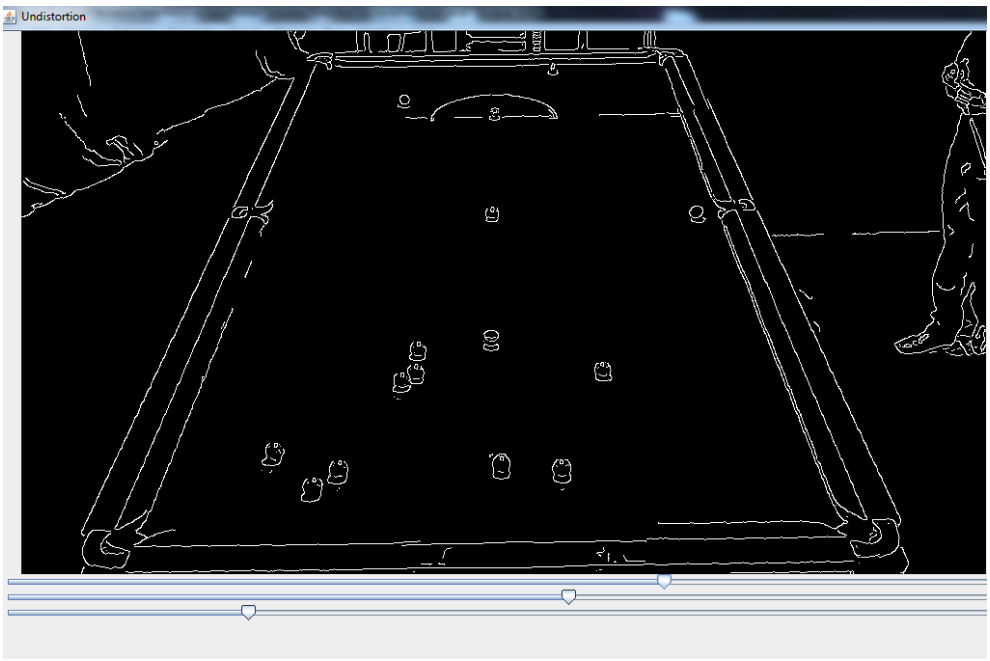
3.1 O zajetih slikah

Vse uporabljene slike so last podjetij Sports Statistics & Information Systems Ltd in World Snooker Ltd, ki sta odstopili pravice do uporabe slik za namen diplomske naloge. Vse slike so bile posnete z mrežno kamero Axis P1347, katere prvotni namen je video nadzor. Nastavljena je bila tako, da vsakih 10 sekund posname sliko resolucije 1920x1080 točk. Slike so bile posnete na dveh različnih turnirjih - World Snooker Championship, ki je potekalo aprila 2013 v Angliji, in European Tour, ki je potekal junija 2013 v Bolgariji. Slike angleškega turnirja so bile posnete na treh različnih mizah, bolgarskega pa na petih. Za namen te diplomske naloge smo uporabili zgolj slike angleškega turnirja.

3.2 Problem popačitve

Vse zajete slike imajo popačeno vertikalno perspektivo, pogosto imenovano „sodček“ (ang. *barrel distortion*), ki nastane zaradi načina delovanja širokokotnih leč [7]. Pri tej distorziji deluje slika napihnjena, napihnjenost slike pa je najbolj opazna pri ravnih linijah. Linija je na sredini izbočena proti robu slike [8]. Za uspešno zaznavo mize in posledično krogel je bila odstranitev

te popačitve prvi korak razvoja našega algoritma. Za ta namen smo razvili grafični vmesnik, ki omogoča nastavljanje moči korekcije popačitve. Vmesnik je prikazan na sliki 3.1.



Slika 3.1: Grafični vmesnik za odpravljanje popačenosti vertikalne perspektive.

Prvi drsnik spreminja moč korekcije popačenja, drugi uravnava povečavo slike, tretji pa prag pri odkrivanju robov, ki ga kasneje uporabimo pri detekciji roba mize (nima pa vpliva na odpravljanje popačitve). Razliko lahko vidimo na slikah 3.2 in 3.3. Da bi poudarili razliko, smo za prikaz na obe sliki narisali ravno rdečo črto. Uporabili smo relativno preprost algoritem za korekcijo popačenja [9].

Slika 3.2: Popačena slika.

Slika 3.3: Popravljena slika.

Poglavje 4

Kalibracija

Za uspešno detekcijo krogel je bilo najprej potrebno določiti območje mize, saj brez informacije o lokaciji mize na sliki nismo mogli določiti lokacij krogel na mizi. Prav tako nam je detekcija mize zmanjšala območje iskanje krogel, kar je omogočilo natančnejše in hitrejše rezultate. Pozicija kamere se skozi set slik ni spreminjala. Tako smo lahko detekcijo mize opravili v sklopu kalibracijskega dela. Pri kalibraciji uporabnik izbere sliko, nad katero algoritem naredi detekcijo mize. Držali smo se pogoste prakse pri odkrivanju robov in smo sliko najprej zgladili ter s tem zmanjšali nivo šuma. Za gladilni filter smo izbrali Gaussov filter [10] z jedrom velikosti 9. Gaussov filter je v praksi zelo pogost, saj se je izkazal kot dobra rešitev za odpravljanje šuma. Empirično smo ugotovili, da velikost jedra 3 in 5, ki sta v praksi sicer bolj pogosta, nista ustrezna, saj je bil nivo šuma v določenih setih slik prevelik. Glajenje z preveliko velikostjo jedra sicer lahko povzroči težave, saj lahko zamegli tudi dejanske robove, a teh problemov pri nas ni bilo, saj je med barvo podlage ter preostankom mize močen kontrast. Z glajenjem slike smo odstranili večino lažnih robov [11].

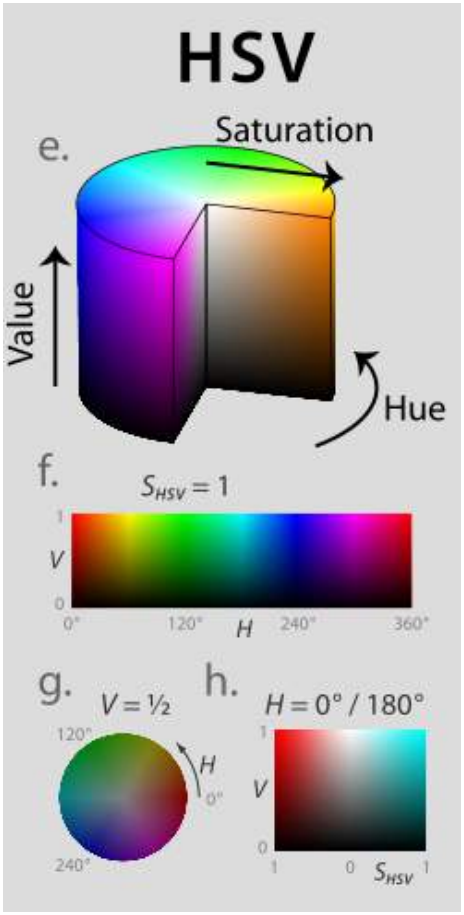
Zglajena slika se najprej pretvori v črnobelo sliko, ki služi kot vhodna slika za Cannyjev algoritem zaznave robov [12], ki je de facto algoritem za odkrivanje robov, saj ponuja zelo dobre rezultate. Mejo za določitev roba lahko uporabnik spreminja na grafičnem vmesniku, kjer se rezultat odkrivanja robov

dinamično posodablja. Kritičen pri detekciji robov je predvsem notranji rob mize, saj je pogosto prevleka položena tudi izven igralne površine. Binarno sliko, ki je rezultat odkrivanja robov, se uporabi kot vhodno sliko za Houghov algoritem odkrivanja črt [15]. Uporabili smo OpenCV implementacijo tega algoritma [16].

Algoritmu je potrebno podati vrednost najmanjše dolžine črte in največji razmak med črtama, preden ju razlikuje kot dve različni črti. Za najmanjšo dolžino smo empirično določili vrednost enako 10% širine slike. Tu smo predpostavili, da bo miza vedno zavzemala večji del površine slike. Za največji razmak med črtama smo vzeli petino vrednosti najmanjše dolžine črte. Algoritem vrne celoten nabor najdenih črt. Iz te množice smo morali izluščiti črte, ki predstavljajo robove mize. Ob predpostavki, da je miza vedno na sredini slike, smo razdelili iskanje na štiri dele. Naš algoritem ločeno poišče zgornji, spodnji, levi in desni rob mize. Pri iskanju zgornjega in spodnjega dela za vse črte izračuna višino (y koordinato) na sredini črte in kot nagiba črte glede na abcisno os. Izbere tisti dve črti, ki sta najbližje sredini slike in ustrezata kotu nagiba. Pri iskanju levega in desnega roba mize pa upošteva x koordinato na sredini črte. Ko algoritem zazna vse štiri črte, izračuna presečišča teh črt in tako določi robove igralne površine. Če zazna premalo ustreznih črt, sporoči napako in odpre okno za ponovitev kalibracijskega postopka.

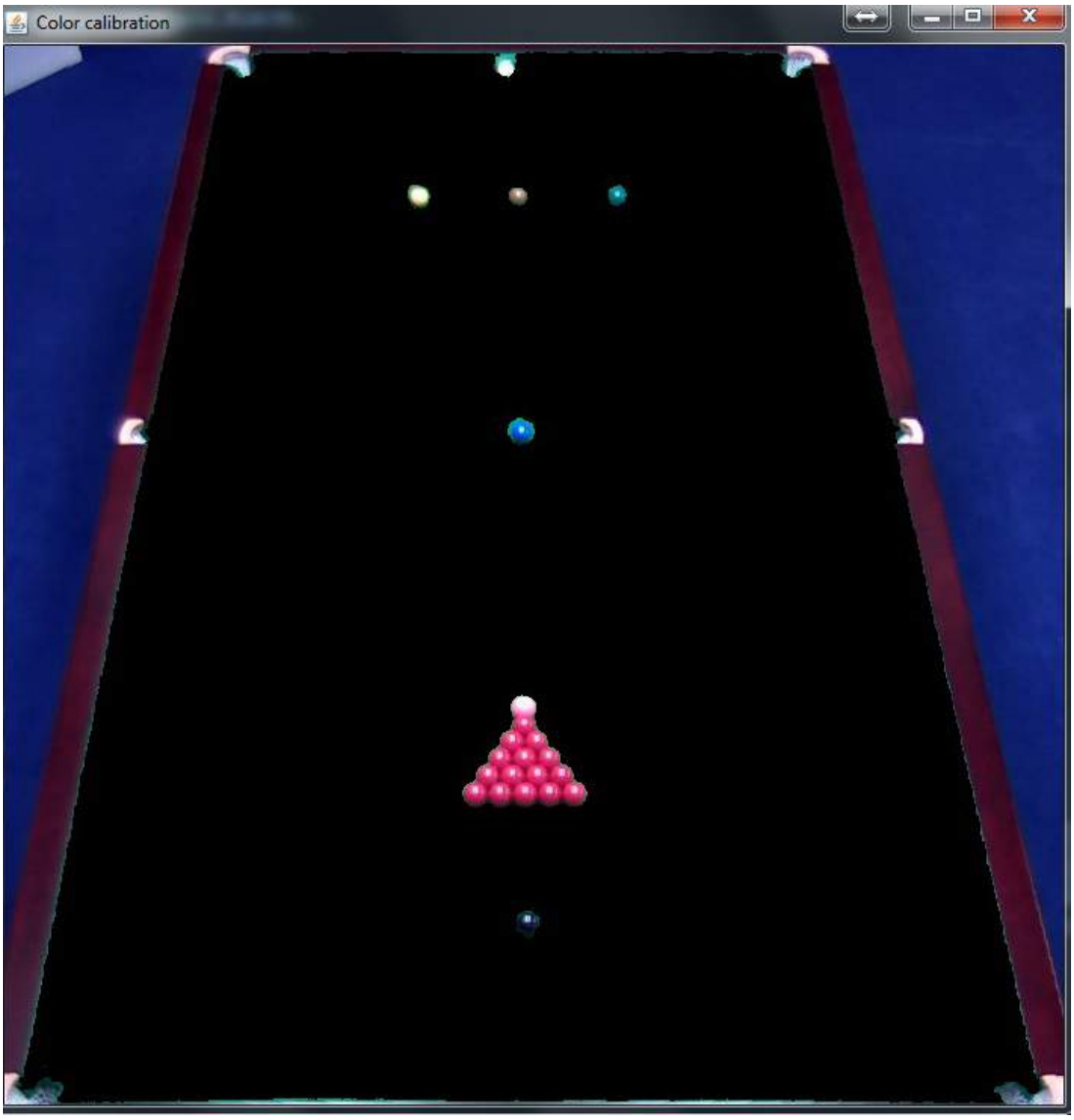
Ko je igralna površina določena, algoritem sliko igralne površine pretvori v barvni model HSV [13]. HSV ali **H**ue(odtenek), **S**aturation(nasičenost), **V**alue(vrednost/svetlost) je v računalniškem vidu pogosto uporabljen barvni model, saj loči informacijo o odtenku barve od ostalih lastnosti. Kako je zgrajen barvni model HSV, smo prikazali na sliki 4.1.

Nato mora uporabnik izbrati točko na igralni površini, ki naj bi predstavljala podlogo. Izbrano točko algoritem izbere kot center kroga z radijem 20 točk. Nad sliko, pretvorjeno v barvni model HSV, izračuna povprečno vrednost kanala H v tem krogu. To vrednost smo privzeli kot povprečno vrednost odtenka podloge. S pomočjo slike HSV izlušči masko podloge mize. To stori tako, da vsako točko slike označi kot del podloge, če ima vrednost H do 7



Slika 4.1: Barvni model HSV.

manjšo ali večjo od izračunane povprečne vrednosti odtenka podlage. Do dovoljenega odstopanja +/- 7 smo prišli s preizkušanjem različnih mej. Nižja dovoljena odstopanja velikokrat niso zadostovala za masko celotne površine, večja pa so občasno zajela tudi dele zelene krogle. Mišljene so 8 bitne vrednosti, torej razpon 256 vrednosti. Zavedati se je treba, da je kanal H za razliko od kanalov V in S cikličen. To pomeni, da sta v OpenCV vrednosti 1 in 254 enako oddaljeni od vrednosti 0, vrednosti 0 in 255 pa predstavljata isti odtenek barve. Slika 4.2 ima čez izvirno sliko s črno barvo narisano masko določene podlage.



Slika 4.2: Slika z narisano masko podlage mize.

Poglavje 5

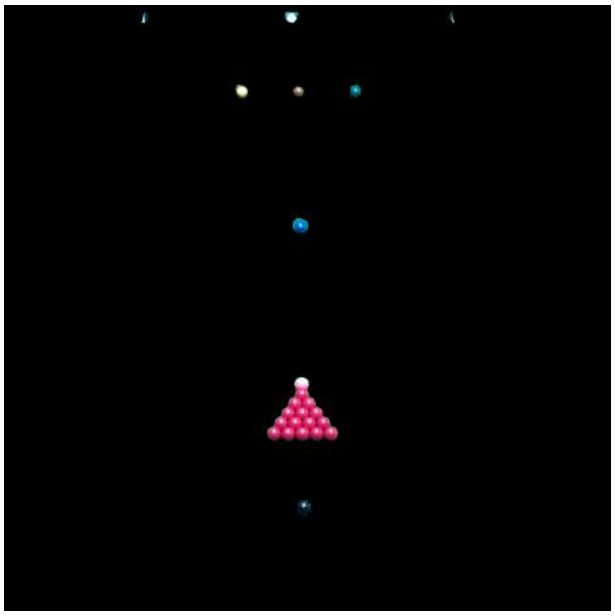
Algoritem iskanja krogel

5.1 Filtriranje slike

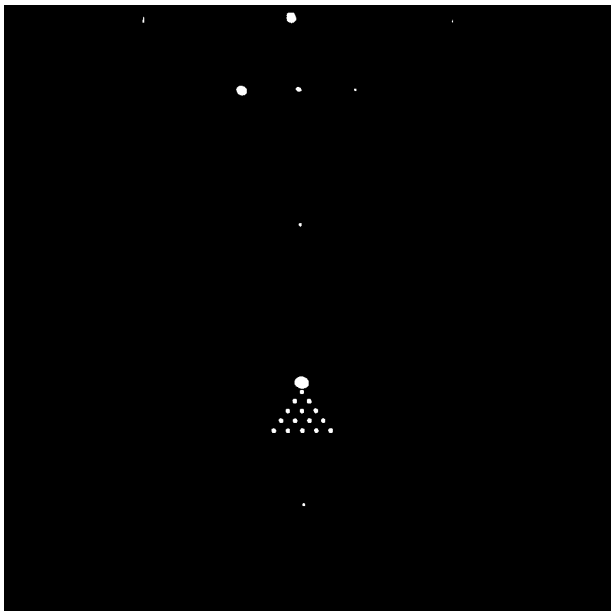
Kot smo omenili, je kalibracija nujen korak, vendar je na srečo potreben le enkrat na vsako postavitev kamere. Ko imamo vse kalibracijske podatke, lahko algoritem nemoteno odkriva krogle. Za vsako sliko, ki jo želimo procesirati, ima algoritem enak postopek. Vhodni sliki najprej opravi korekcijo popačitve slike s parametri, ki jih je uporabnik pred tem izbral v procesu kalibracije. Sliko nato obreže okoli izračunanih mej mize, tako da ostane le igralna površina. To sliko pretvori v prostor HSV in z enakim postopkom kot pri kalibraciji odstrani igralno podlogo. Ostane nam slika podobna sliki 5.1. To sliko nato pretvorimo v črnobelo sliko. Za vsak piksel črnobele slike preverimo vrednost piksla; če je vrednost večja kot 150 (od največ 255), mu spremenimo vrednost na 1, v nasprotnem primeru pa na 0. Tako dobimo masko, ki predstavlja svetlejše dele slike. Primer maske je na sliki 5.2. Tudi do te meje smo prišli s preizkušanjem različnih vrednosti.

5.2 Določitev področja zanimanja

Nad masko poženemo algoritem iskanja kontur. Kontura je krivulja, ki povezuje vse neprekinjene točke enake barve ali intenzitete. Tako dobimo obris



Slika 5.1: Filtrirana slika.

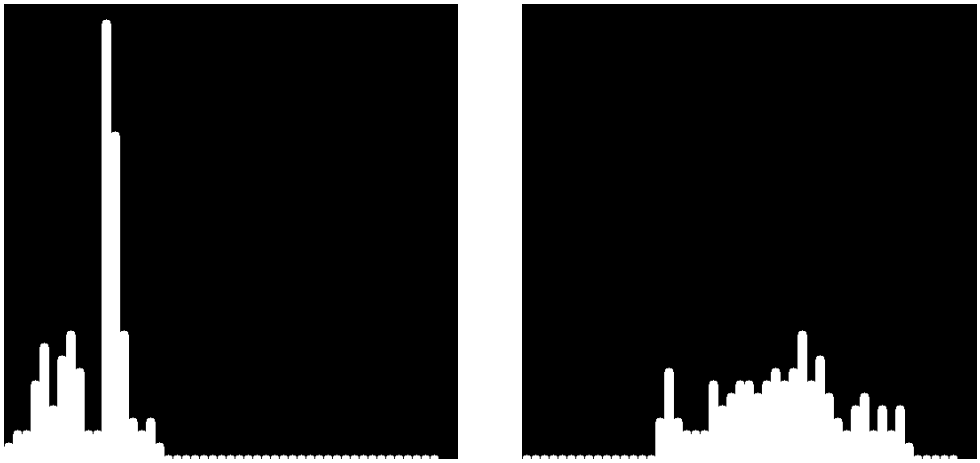


Slika 5.2: Maska svetlejših površin.

predmeta. V OpenCV je ta algoritem že implementiran [14]. Izmed vseh najdenih kontur izločimo tiste, ki imajo preveliko površino. Če je površina konture večja od površine največjega možnega kroga, potem očitno ne more predstavljati krogle in jo odstranimo. Preostale konture nam služijo za določitev področja zanimanja (ROI – **R**egion **o**f **I**nterest). Za vsako konturo nam OpenCV poda tudi informacijo o poziciji središča konture na sliki. To točko vzamemo kot središče kvadrata, ki ima stranice dvakrat daljše kot je premer največjega možnega kroga v tej točki. Ta kvadrat je naše področje zanimanja. V vsakem kvadratu iščemo točno eno kroglo. Področje zanimanja postavimo na sliko, na kateri so rezultati odkrivanja robov.

5.3 Histogram oddaljenosti

Za vsak piksel v tem kvadratu izračunamo evklidsko razdaljo za vse ostale piksle, ki imajo vrednost 1. Oddaljenost zaokrožimo na celo število, ki ima tako lahko vrednost med 1 in dolžino diagonale kvadrata. S temi vrednostmi gradimo histogram razdalj, kjer dolžina deluje kot indeks histograma. Za vsako izračunano razdaljo za 1 povečamo vrednost histograma pri tej dolžini. Tako za vsako točko dobimo histogram oddaljenosti vseh neničelnih točk.



(a) Histogram piksla na sredini kroga. (b) Histogram piksla na robu slike.

Slika 5.3: Primerjava histogramov.

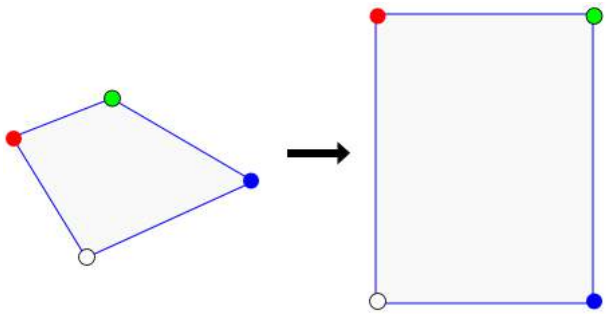
5.4 Odločanje o izidu

Na podlagi histograma se odločimo, ali se v trenutnem kvadratu lahko nahaja krogla. Tu izkoriščamo dejstvo, da bo imel piksel na sredini kroga ekstrem enako oddaljenih pikslov. Na sliki 5.3 vidimo primerjavo histogama piksla, ki je na sredini kroga, in histograma piksla, ki je na robu kvadrata. Če smo

Slika 5.4: Zaznane krogle.

v območju zanimanja našli krog, ga shranimo in nadaljujemo, dokler ne preiščemo vseh območij zanimanja. S tem smo določili vse krogle na trenutni sliki. Slika 5.4 ima čez originalno sliko z vijolično barvo narisane zaznane krogle.

Narisane krogle imajo velikost, kakršno je določil algoritem. Kot lahko vidimo, so pozicije in velikosti krogel praktično identične dejanskim. Vendar na tej stopnji nimamo podatka o lokaciji krogle na mizi, temveč zgolj podatek o lokaciji krogle na sliki. Da bi bile te lokacije krogle na sliki uporabne, moramo določiti lokacijo krogle glede na igralno mizo. Ker pa miza ni posneta iz ptičje perspektive, je glede na oddaljenost od kamere miza različno široka. Sliko moramo zato najprej preslikati v ptičjo perspektivo [18]. OpenCV ima algoritem za perspektivno preslikavo že implementiran. Algoritmu je potrebno podati 4 vhodne in 4 izhodne točke; vhodne točke služijo kot vhodna oblika, ki jo želimo preslikati, izhodne štiri pa kot zaželjena oblika. S temi osmimi točkami algoritem izračuna homografijsko matriko, ki definira projekcijsko transformacijo. To matriko podamo skupaj z vhodno sliko, da dobimo transformirano sliko. Te algoritme pogosto uporabljamo za preslikavo v ptičjo perspektivo, kot je prikazano na sliki 5.5.



Slika 5.5: Homografija.

V našem primeru so vhodne štiri točke robovi mize, ki smo jih že določili. Snooker miza je po turnirskih standardih dolga 357cm in široka 178cm. Razmerje dolžino in širino je tako približno 2. Za izhodne točke podamo pravokotnik, ki ima enako razmerje stranic. Na sliki 5.6 je levo slika pred perspektivno preslikavo, desno pa izluščena miza v ptičji perspektivi.

Po enakem principu, kot algoritem preslika točke, lahko s pomočjo homografijske matrike izračunamo tudi položaje središč krogel na preslikani sliki. Homografijska matrika je velikosti 3x3 in vsebuje realna števila. Nove koor-

(a) Vhodna slika

(b) Miza v ptičji perspektivi

Slika 5.6: Preslikava mize.

dinate točke izračunamo s pomočjo formul 5.1 in 5.2. Števila a, b, ..., h, 1 pripadajo homografijski matriki, ki je vidna na sliki 5.7.

$$x_1 = \frac{a \cdot x + b \cdot y + c}{g \cdot x + h \cdot y + 1} \tag{5.1}$$

$$y_1 = \frac{d \cdot x + e \cdot y + f}{g \cdot x + h \cdot y + 1} \tag{5.2}$$

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{bmatrix}$$

Slika 5.7: Homografijska matrika

S določitvijo x_1 ter y_1 smo dobili informacijo o točni poziciji krogle in s tem dosegli cilj diplomske naloge.

Poglavje 6

Rezultati

6.1 Rezultati na množici slik

Za testiranje našega algoritma smo izbrali 300 slik iz treh različnih miz, kjer vsako mizo predstavlja 100 slik. Ocenjevanje uspešnosti in natančnosti je potekalo ročno. Za vsako najdeno kroglo smo na vhodno sliko z vijolično barvo narisali krog s centrom v točki, ki jo je določil algoritem in z določeno velikostjo krogle v tej točki. Nato smo opravili vizualni pregled odstopanj velikosti in pozicij. Za popolnoma pravilne ocene rezultatov bi morali klikniti na sredino krogle pri vseh možnih kroglah in ročno vnesti velikost. Vseh krogel je skoraj 5000 in tako preveč za tak pristop. Prav tako bi morali razviti grafični vmesnik, ki bi omogočal ročni vnos. Tudi določanje sredine krogel zahteva veliko povečavo slike in zelo natančen klik na sliko. Menimo, da je vizualna ocena dovolj natančna za potrebe te diplomske naloge. Na izbranih 300 vhodnih slikah smo našteli 4977 krogel. Slika je tako v povprečju vsebovala približno 16,6 krogel. Naš cilj je bil zaznati čim več krogel. Zaradi precejšnega števila določljivih primerov smo se odločili, da raje označimo več pravilnih krogel za ceno malo večjega števila lažnih krogel. Naš algoritem smo temu prilagodili in je uspešno odkril 4962 krogel ali 99,7% vseh možnih. Skupaj je torej zgrešil 15 krogel. Nikoli ni zgrešil več kot eno kroglo, vse krogle pa je pravilno določil na 285 slikah ali 95% vseh slik. Na sliki 6.1

lahko vidimo primer, ko algoritem ni uspešno zaznal črne krogle. V večini teh primerov je bila za zgrešeno detekcijo kriva močna senca pod kroglo, ki je tudi glajenje slike ne odpravi popolnoma. Lažnih krogel smo našteli 165,

Slika 6.1: Manjkajoča krogla.

torej je algoritem dodal 3,3% neobstoječih krogel. Dodatne lažne krogle ima 74 slik. Teh 74 slik ima tako povprečno 2,23 krogle preveč. Največja napaka na posamezni sliki je bila 7 dodatnih krogel. Večino lažnih krogel je algoritem zaznal na igralcih, ki so sloneli čez mizo, v nekaj primerih pa je kot krogle označil tudi dele igralne palice. Primer, ko je algoritem zaznal igralčevo roko kot kroglo, je na sliki 6.2.

Rezultati bi bili precej boljši, če bi delali analize pozicij krogel na slikah z neoviranim pogledom na mizo. Pet slik ima obenem mankajoče in dodatne krogle. Brez napak števila krogel je tako 221 slik oziroma 73,67% vseh. Ne smemo pozabiti, da bi lahko krogle v nadaljevanju izločili tudi na podlagi

Slika 6.2: Napačno zaznana krogla.

barve, čeprav to ne bi bilo učinkovito, če bi potencialno kroglo izločil algoritem za odkrivanje krogov. Menimo, da je tako odstotek vseh odkritih krogel pomembnejši. Seveda pa je detekcija krogel brez napak cilj, h kateremu stremimo. Na sliki 6.3 je primerek pogoste zaznave krogel. Določeni centri krogel ne odstopajo veliko od dejanskih. Na tej sliki je še najbolj zgrešena pozicija zelene krogle desno zgoraj. Največja napaka ocenjene velikosti je pri beli krogli, kar je v veliki meri posledica dejstva, da je bela krogla dejansko večja od ostalih, naš algoritem pa se ne zaveda razlik v barvah krogel. Prav tako nam je od velikosti pomembnejša pozicija krogle. Če bi na primer hoteli narisati krogle v ptičji perspektivi v nekakšnem grafičnem prikazu stanja na mizi, bi si velikost krogle izbrali sami. Napaka pozicije krogle ni bila nikjer večja od velikosti krogle. Večina položajev krogel pa je določena do treh pikslov natančno. Na skoraj 4 krat 2 metra veliki mizi to pomeni centimetrsko

Slika 6.3: Tipična zaznava.

natančnost.

6.2 Časovna zmogljivost

Časovne meritve algoritma smo izvajali na Core 2 Quad procesorju, ki deluje s frekvenco 2660 MHz. Merili smo čas, ki ga algoritem potrebuje za celotno analizo slike in določitev vseh krogel. Izmerjeni povprečni čas za analizo slike je bil 425 ms, od tega je za samo iskanje krogov v območju zanimanja porabil v povprečju 150 ms. Preostali del časa so zavzemale različne obdelave slike. Ta del je fiksen, iskanje krogov pa narašča s kvadratno časovno zahtevnostjo. V kvadratu velikosti n moramo za vse točke izračunati dolžine vseh ostalih neničelnih točk. Ker so območja zanimanja relativno majhna, kvadratna časovna zahtevnost ni bil problem. Doseženi časi nam omogočajo obdelavo

dveh slik na sekundo, na novejšem procesorju pa bi jih lahko obdelali še več. Za našo pogostost zajetja slike na 10 sekund je algoritem je več kot dovolj hiter tudi na našem procesorju. Brez težav bi v realnem času obdelali tudi slike, posnete vsako sekundo. Za sledenje kroglam na video posnetkih, ki imajo ponavadi med 25 in 30 sličic na sekundo, bi morali naš algoritem dodatno optimizirati.

6.3 Primerjava našega algoritma s detekcijo krogov s Houghovo transformacijo

V računalniškem vidu je Houghova transformacija izjemno priljubljena za detekcijo različnih oblik [17]. Enako kot naš algoritem običajno operira nad sliko odkritih robov. V primeru, da iščemo krog, vsako točko, ki predstavlja rob, iz (x, y) prostora preslikamo v (x_0, y_0, R) prostor. V tem prostoru x_0 ter y_0 predstavlja središče kroga, R pa njegov radij. V našem primeru radij poznamo, tako da se dimenzija parametrov poenostavi na 2. Za vsako točko x in y določimo vse kroge s tem radijem, ki vsebujejo to točko. Vsak določen krog služi kot glas. Točka v preslikanem prostoru, ki ima največ glasov, ima največjo verjetnost, da je krog. Problem iskanja oblike tako pretovorimo v iskanje maksimuma števila glasov.

OpenCV ima Houghovo transformacijo za iskanje krogov že implementirano. Na začetku smo za iskanje krogel uporabljali to implementacijo, ki pa se je na žalost izkazala za premalo zanesljivo rešitev. Pri tej implementaciji je potrebno podati pričakovane polmere krogov in prag glasov, potrebnih za sprejetje kroga. Poskušali smo z več načini določanja praga, vendar se nobeden od njih ni izkazal za dovolj dobrega - velikokrat smo dobili preveč lažnih krogov, če pa smo prag zaostrili, pa je izločil preveč dejanskih krogov. Sama hitrost iskanja krogel je sicer večja in bi omogočala večje število obdelanih slik.

Poglavje 7

Sklepne ugotovitve

Opisali smo vhodne podatke in predstavili kalibracijski postopek. Algoritem smo poskušali predstaviti brez poglabljanj v detajle samega delovanja različnih delov že implementiranih algoritmov knjižnice OpenCV. Prav tako smo izpustili različne detajle naše implementacije, ki niso pomembni pri osnovni ideji algoritma. V primerjavi z Houghovo transformacijo za iskanje krogov smo z lastnim algoritmom za detekcijo krogov izjemno izboljšali rezultate. Žrtvovali smo sicer nekaj hitrosti, vendar je bila ključna pridobitev našega algoritma veliko večja natančnost. Glavni del te diplomske naloge je bil programski del, ki nam je vzel precej več časa, kot smo prvotno ocenili. Prav tako smo se morali pogosto poglobiti v različne implementacije algoritmov, saj je dokumentacija OpenCV pogosto pomanjkljiva in ne posreduje zadostne količine informacij o že implementiranih algoritmih v tej knjižnici. Osnovna detekcija je bila hitro dosežena, nato pa je bilo potrebno veliko popravkov, da smo dosegli zadovoljive rezultate. Ti so veliko boljši, kot je bila prvotna implementacija, vsekakor pa je še prostor za izboljšave. Največ se lahko še pridobi na zmanjšanju števila zaznanih lažnih krogel. Uporabna vrednost bi tudi precej zrasla s določanjem barv zaznanih krogel. Dosegli smo skoraj 100% detekcijo krogel pri 3,3% lažnih kroglah, kar ocenjujemo kot dober rezultat. Nekateri podobni projekti so dosegali podobne ali celo slabše rezultate, res pa je, da so povečini poskušali določiti še barve. Naš projekt je

odlična osnova za nadaljnji korak detekcije barv, ki pa je za obseg našega diplomskega dela enostavno prevelik.

Literatura

[1] Glavna stran knjižnice OpenCV
http://opencv.org/

[2] Splošno o snookerju
http://en.wikipedia.org/wiki/Snooker

[3] Zgodovina snookerja
http://www.worldsnooker.com/page/AboutHistoryofSnooker

[4] Zgodovina snookerja
http://www.snookerclubs.com/history-of-snooker.htm

[5] Zgodovina snookerja
http://en.wikipedia.org/wiki/History_of_snooker

[6] Največje število doseženih točk
http://en.wikipedia.org/wiki/Maximum_break

[7] Odpravljanje napak leč in objektivov
http://www.slo-foto.net/novice/2010/09/21/
kako-v-postprodukciji-popraviti-oz-odpraviti-napake-lec-in-objektivov

[8] Efekt ribjega očesa
http://stackoverflow.com/questions/2477774/
correcting-fisheye-distortion-programmatically

[9] Algoritem za odpravljanje popačitve
http://stackoverflow.com/a/2502276

[10] Gaussov gladilni filter
http://en.wikipedia.org/wiki/Gaussian_blur

[11] Odkrivanje robov
http://en.wikipedia.org/wiki/Edge_detection

[12] Cannyjev algoritem
http://www.lakos.fs.uni-lj.si/attachments/article/204/poročilo%20seminar.pdf

[13] Barvni model HSV
http://en.wikipedia.org/wiki/HSL_and_HSV

[14] Odkrivanje kontur
http://docs.opencv.org/doc/tutorials/imgproc/shapedescriptors/find_contours/find_contours.html

[15] Houghova transformacija
http://homepages.inf.ed.ac.uk/rbf/HIPR2/hough.htm

[16] OpenCV implementacija za iskanje črt
http://docs.opencv.org/doc/tutorials/imgproc/imgtrans/hough_lines/hough_lines.html

[17] Houghov algoritem za odkrivanje krogov
http://en.wikipedia.org/wiki/Hough_transform

[18] Podbreznik P. (2011) *Vzpostavitev korespondence pri velikih odmikih dveh kamer za poljubno izbrano točko kot prispevek k izboljšanju razpoznavanja slik.* Doktorska disertacija, Maribor : Fakulteta za elektrotehniko, računalništvo in informatiko